Interactive Operations for Visualization of Ad-hoc Sensor System Domains

Mark A. Livingston and Evan V. Herbst
Virtual Reality Laboratory
Naval Research Laboratory
Washington, D.C.
markl@ait.nrl.navy.mil, herbst@ait.nrl.navy.mil

***Abstract*— Embedded sensors are envisioned to provide information about a variety of environments. One use of sensors is for military intelligence gathering in the field. Typically, these sensors are manually placed in an environment. In order to know whether the appropriate domain is covered by the sensor arrangement, we build an interface that provides visualizations of the sensor domains and coverage. To assist with planning the sensor layout, our new interface allows the user to interactively specify new locations and see the effect on the domain. To verify that line-of-sight communications requirements are met, the visualizations also show the user these relationships. These simple visualizations improve the understanding of the sensor domain. The interactive 3D display gives the user a tool for planning how to best use the available resources to get information.**

I. Introduction

Embedded sensors have the potential to provide users vast amounts of information about the environment. Proposed applications span a wide range, including seismic monitoring [1] to warn of potential natural disasters, monitoring of oil and gas pipelines for capacity and oil fields for remaining reserves [2], ecological monitoring of habitats [3] or glaciers [4], and monitoring homes for the safety and health of its inhabitants [5]. But in order to gather the right information, the sensors must be placed in the environment in a way that enables the network as a whole to see the entire area of interest and avoid areas that will give rise to errors in the sensor readings.

In many situations, the sensors and associated databases and processors are only capable of delivering low-level, unstructured data such as raw measurements. If users are to be able to extract meaning from the stream of data, interfaces must be provided to allow users to form and execute queries. But just getting a value may not be enough; in some applications, it is important to know other information about the data, such as from where the raw measurements were gathered. Online query processing [7], [8] could enable users to understand the scope of reported data. For our application environment, however, we wanted a tool that could assist with planning the sensor arrangement before sending someone out in the field.

We describe our implementation of a graphical interface that assists a user in planning the arrangement of sensors in an environment. Using terrain information and sensor characteristics, we compute and display line-of-sight relationships between sensors and domain coverage masks, such as shown in Figure 1. These two pieces of information are important for our intended application. The operating environment for our application makes proper planning before setting out to place sensors in the environment critical.

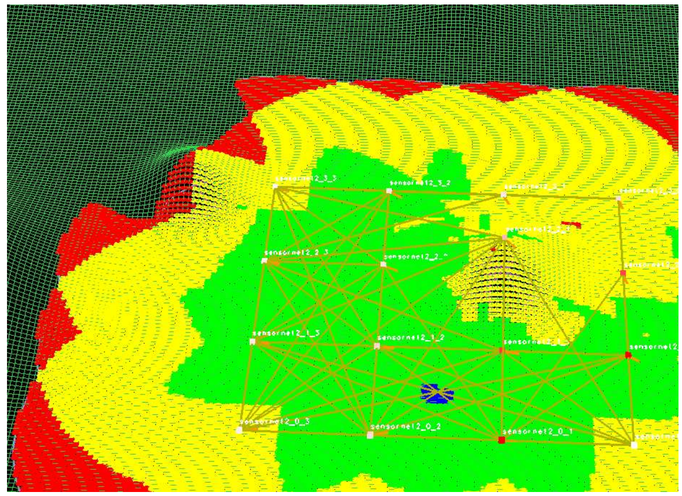

Fig. 1. A snapshot from the interactive system for planning arrangement of sensors in an environment.

II. Problem Domain

We consider a military use of sensors: providing early warning of movement toward friendly positions. We consulted with domain experts and the relevant Marine Corps doctrinal publication [9] to determine the most pressing limitations with current interfaces. We note the following issues.

1) **Terrain masking**.
   Current sensor suites require that line-of-sight contact be maintained (possibly through relays) to each sensor. These relationships are not easy to visualize in complex terrain. Redundancy may increase the connectivity to the sensor network. Relay stations may be set up in order to establish connections to all deployed sensors.
2) **Sensor positioning**.
   It is hard to place sensors in such a way that achieves coverage of the desired area. Limited resources may preclude covering all of the area of interest.

With practice, anyone could develop 3D visualization skills that would make these operations easier. However, our primary goal was to produce an interface that would make these operations intuitive for novice users and in environments of arbitrary geometric complexity. We can assume that a detailed model of the terrain is available in advance of the planning of sensor locations.

It should be noted that another hard problem is understanding what the data tells the user while the sensors return valid data (above the level of noise). Currently, the user views a primitive interface that shows sensor outputs on dial-shaped or numerical displays; the user must integrate these values into a coherent picture to determine what action to take. While this is an obvious area for improvement, it is beyond the current scope of our project.

Note that this is not a problem domain in which one can reasonably expect to adjust the sensor arrangement repeatedly. Entering the environment can be dangerous; at a minimum, it potentially reveals one's actions and intentions, both of which are consequences to avoid. Thus a secondary goal for such a tool is to reduce the number of in-situ adjustments necessary and to increase the speed with which initial placement and any in-situ adjustments are performed.

We divide the types of sensors that could be applied to this problem into two categories by the geometry of their domain. First are omni-directional sensors, such as magnetic or vibration sensors. Seismic sensors tend to have a longer range than magnetic sensors, but the range of seismic varies with soil type. Typical values for detection are 100 meters for vehicles and 25 meters for people. Seismic sensors also detect a variety of natural phenomena, such as water flow in rivers or at the shore or volcanic activity, in addition to their military application of detecting ground vibrations caused by people or vehicles. Magnetic sensors generate a magnetic field and detect disturbances caused by ferrous metals; they can determine direction of movement across the field. Power lines, stationary metal objects, and metallic ores in the ground can interfere with magnetic sensors without proper accounting for the ambient electromagnetic field. Typical ranges for detection are 25 meters for vehicles and 3 meters for people. These types of sensors are frequently attached to each other to form strings of a few sensors to be laid out in a line.

The second class of sensors from a geometric standpoint are directional sensors, such as infrared imaging devices. Their range is comparable to seismic sensors, but of course requires line-of-sight contact for any object to be detected. Relays that are currently available are ground-based. All of these sensors are used by current units that employ sensing technologies in the field. Future enhancements may include airborne relays, improved sensors, and new types of sensors, but we do not want to design an interface that relies on these advancements being available. Fortunately, many of these improvements are orthogonal issues to those in the interface, requiring different parameters (e.g. range), but not fundamentally different models of sensors.

III. Visualization System

Our visualization system begins by loading a simple terrain elevation model. We provide six degree-of-freedom control of the viewpoint with a mouse-based interface [10] composed of orbit, dolly, truck, and roll operations. However, roll would seem to be unnecessary, since it is more natural to keep the world up vector vertical on the screen. Having interactive control of the viewpoint provides the user with motion parallax, helping them interpret the shape of the environment more quickly and more accurately. The system next loads sensor characteristics and locations of such sensors or strings of such sensors in the environment. The program then computes the initial for the operations determined in our analysis of the problem domain to be high-priority needs. The system is designed to be extensible, with sensor characteristics given to the program at run-time. (Sensor models must be programmed into the system.)

A. Environment Model

The terrain model consists of a height field sampled on a regular grid. At each vertex, we compute the normal; this is used for terrain analysis and for rendering quality. For each vertex, we store the following information found in the Compact Terrain Database (CTDB) format.

- U.S. Geological Survey soil category: a discrete value that indicates various types of rock, stone, soil, mud, sand. These are separated into five categories that indicate how much they amplify or transmit seismic activity.
- Surface material from IEEE standard 1278.1. We are primarily interested in the water, grass, and road types.
- Soil moisture content: a discrete value from the set { unknown, dry, moist, wet, frozen }.
- SIMNET and CCTT mobility ratings for vehicles and pedestrians through the terrain type. These are integer values in the range of 0-15 and 0-31, respectively. These are unused in our system.
- Boolean descriptive values: water, road, slow-going, impassable. These are also unused in our system.

Though the Boolean values are derivable from the other values, the input is not checked for consistency; the Boolean values are read from the file. The visualizations and operations implemented thus far use only the first three variables; the others are included for compatibility of format and potential future use.

The terrain elevation data may be given as a grid of points or by a function. The terrain characteristics are given in the same system (grid or function) as the elevation data. In the case of a function, a gridded domain for the function is given, and the elevation function should evaluate to a floating-point value at every point in that grid. The terrain characteristic functions are preceded by a type (e.g. a value for the surface material); they must be Boolean functions over the same domain as the terrain model. For any grid point, a value of "true" indicates that the point is of the type that preceded the function. Each new function can overwrite the output of previous functions. This terrain model is rendered as a quadrilateral mesh in either wireframe or filled polygons.

This terrain visualization allows the user to see features of the environment that could be of interest in the task of sensor arrangement, such as rivers and roads (Figure 2). Flowing water is an example of an environmental effect on certain sensor types; thus it is an important feature when planning

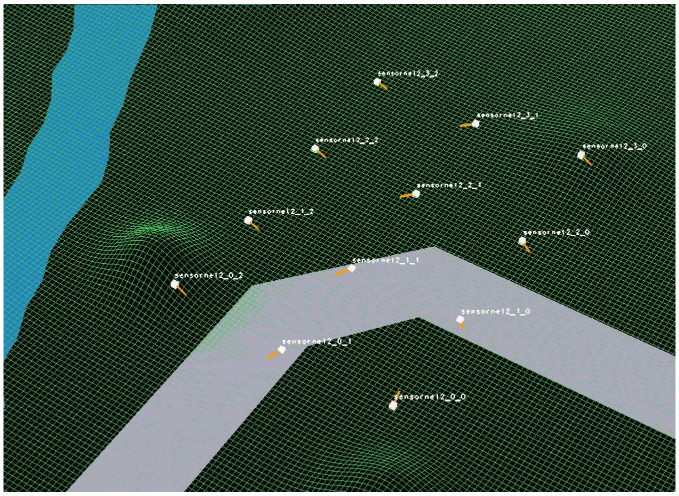

Fig. 2. A sample environment in which a user places sensors.

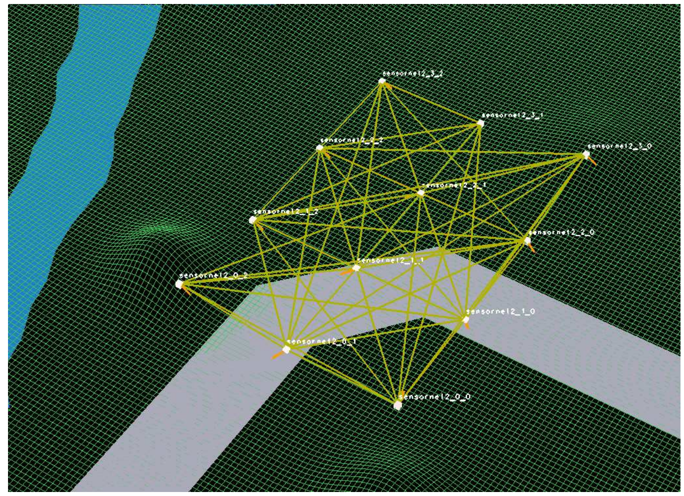

Fig. 3. Line-of-sight relationships in the sample environment. Note the redundancy in the possible connections.

sensor positions. A road is a feature which has a semantic meaning for the user; this is a likely location to find a vehicle, which is a primary purpose of our users' application. An automated terrain analysis can highlight areas that might be of strategic interest, such as the bases of hills or passes between hills that might be easier to traverse than the summit of a hill. Such features are detected with differential geometry operators, which measure curvature and find zero-crossings or values over a threshold to determine when the terrain changes from a hill or valley to a (nearly) flat patch. These operators use the terrain normals as input.

B. Line-of-sight Relationships

To determine the masking effect of the terrain on line-of-sight communications capabilities, we use ray shooting queries [11] from each proposed sensor location to each other proposed sensor location. The rays are intersected with the terrain model. Any intersection that occurs closer to the initiating sensor than the receiving sensor indicates that line-of-sight contact does not exist between those two sensors. This is a symmetric query, a fact we can use to reduce the amount of computation. We use the fact that sensors are grouped into strings to reduce the number of queries. If, for example, each adjacent pair of sensors has line-of-sight contact, then we need not test non-adjacent sensors within a single string; the entire string can communicate. Note that this implies that not all individual sensors must have line-of-sight contact with all other sensors along a string. A general spatial subdivision algorithm [12] also serves to reduce computation. If a ray does not enter a bounding region (whether due to a closer intersection or the lack of an intersection), no further queries to sensors in that region should be performed.

For each successful connection found by a ray query, we draw a line connecting the two sensor locations (Figure 3). While this can result in a busy display, it is important to know that redundancy exists in the network connections possible between sensors. Thus the user may plan for possibilities of damage due to exposure to weather, vehicles, and pedestrians, or of having a sensor discovered and removed. A sensor may be disabled from the computations to simulate these conditions; new queries might be needed to re-establish the connectivity relationships. For example, if a sensor is in the middle of a string, its adjacent sensors should now be checked to establish whether connectivity exists over the string.

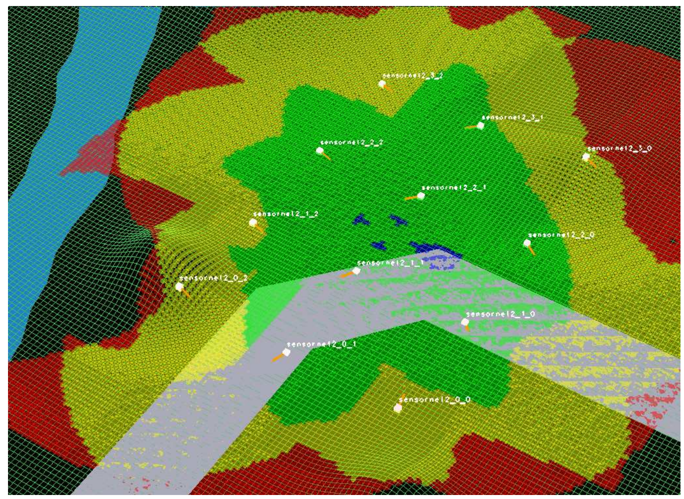

Fig. 4. Domain coverage of the network in the sample environment. The color gives the number of sensors that can see each region: red=1, yellow=2, green=3, and blue=4.

C. Domain Coverage

The number of sensors that see a particular point in the environment helps a user know not merely whether the various regions of the environment are covered, but also the reliability of information about that region. This is crucial information for military applications. The domains of each sensor are integrated into a single visualization of the domain of the entire network (Figure 4).

The computation of the coverage mask accounts for the interaction of the sensor with the terrain. Recall that vibrations propagate at different rates based on density of the ground material. This and the terrain features account for the irregular shape of the combined domain as seen in Figure 4. To determine the extent of a sensor's domain, we propagate the sensor with a breadth-first search of the vertices in the mesh, accounting for the soil type and distance from the sensor.

D. Sensor Arrangement

The initial arrangement of sensors (and characteristics of those sensors) within the environment is read upon startup. To become an interactive planning tool, the interface must allow a user to specify new locations for sensors. The user can specify a location for an endpoint of a string of sensors with a simple click on the environment. The program constrains the sensor to be on the ground or at a predetermined height with respect to it. This constraint along with the 2D position on the user's view of the environment at which the mouse click occured gives a 3D position for the sensor. In cases where there are multiple intersections of the ray with the terrain, the nearest intersection is taken. Since the user has full viewpoint control, this is a natural constraint to impose.

For a string of sensors, the user can specify a second location in one of two ways. The second location may be given manually (by clicking on the location, just as for the first location). In this case, the system will create a string with a number of sensors given through user interface widgets and using the direction and spacing from the first to the second sensor. The string may also be created parallel to the coordinate axes that are assigned to the environment. This gives the direction in which to move in 2D; the location must be lifted to the proper height in the same manner when the user enters the location with the mouse. The user gives the distance between sensors (i.e. distance along the string) and number of sensors in a string through user interface widgets. Since all military maps come with a coordinate system, we may assume we have this coordinate system for such an operation. Figure 5 shows the new sensor arrangement, line-of-sight relationships, and coverage mask for the sample environment after a string has been added to the arrangement shown in Figures 2–4.

The program also allows the user to request a suggested sensor placement that would maximize the area added to a network of sensors under the constraint that the new sensor must be able to communicate with the existing network. This is done by an iterative search of directions from the current network in which there are no sensors. Sensors may be deactivated from the domain computations to simulate loss of a sensor in the network.

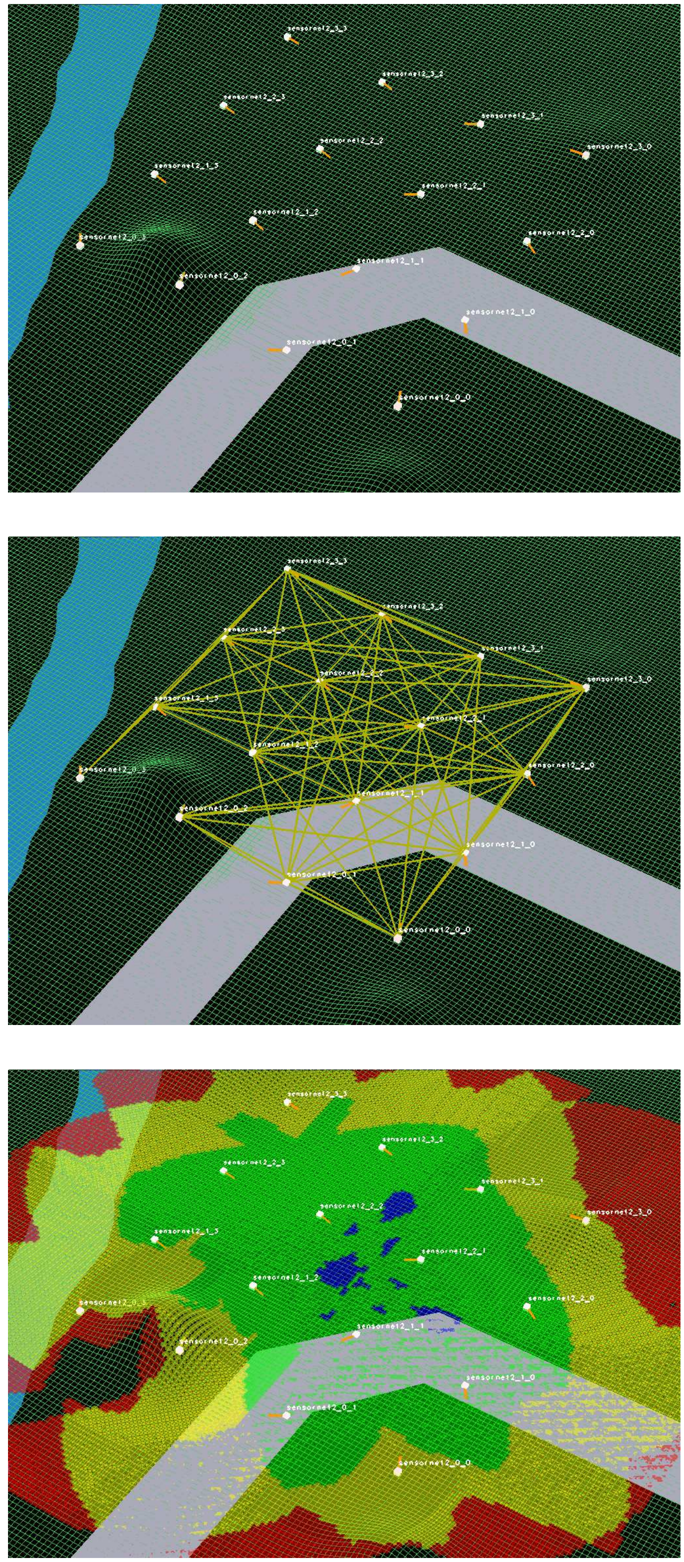

Fig. 5. Results of an interactive placement of new sensor locations. *(top)* The user assigns the first location (far left) and through a widget panel (not pictured) that the string should follow the coordinate X axis, towards the upper right in this image (cf. Figure 2). *(middle)* The new line-of-sight relationships show that the sensor on the far left has connections in only one direction, a tenuous connection to the remainder of the network (cf. Figure 3). *(bottom)* The new coverage mask shows that the stream on the left is well within the domain of many of the sensors (cf. Figure 4), which may cause the seismic readings to be noisy, depending on the strength of the stream.

IV. Discussion and Future Work

We developed visualizations that should help plan the arrangement of sensors for military applications. The system is not in use, but initial reactions from domain experts are positive. Once integrated into existing systems in the field, this interface should reduce some important limitations encountered when using sensors for detection of movement towards friendly positions. The two primary advantages are knowing whether line-of-sight relationships exist so that devices may communicate to the data server, and whether a given area of the environment is covered by an appropriate number of sensors. Note that these tools are not fully automatic, nor do they need to be. These operations are controlled by the human operator. The interface is there to assist in the task. The goal is to increase the speed in planning sensor layouts and reduce the risk that the human operators would have to incur in arranging the sensors in the way that best makes use of their available resources.

The problems that we need to solve in order to present a usable system may be solved with simple algorithms based on geometric queries and graph algorithms. The geometric operations may be grouped into area-based operations (coverage within a domain) and ray-based operations (line-of-sight queries). We also grouped the sensors by the geometry of their domain. To this point, we have only solved the problem for devices with omni-directional domains. As noted above, the sensor suite currently in use includes imaging devices. Integrating such directional sensors into the domain and coverage queries complicates the geometric problems, requiring the integration of qualitatively different shapes into a single geometric description. However, the graph-based propagation algorithm can easily be adapated by augmenting the intersection operation with the sensor domain from the distance-based query used for omni-directional sensors to a query based on distance and direction.

We would like to extend our efforts to show the results of detection algorithms running on the data gathered from these sensors. This requires integration of our interface with a tracking algorithm [14]. Some of the geometric operations, notably the domain computations, are somewhat slow. One of the most expensive operations is computing the coverage mask. While the graph-based approach improves over an early implementation that used geometric intersection of the 2D domain shape and the height field, the current implementation is still slower than we would like. It is also dependent on the resolution of the terrain grid, a feature we would like to avoid. We are considering ways to reduce the computational complexity of the geometric algorithms.

This simple interface improves the understanding of the sensor domain. The interactive 3D display for an application that had not previously seen such a display gives the user a tool for planning how to best use the resources available to get information. The interactions themselves require nothing more than simple desktop or mobile systems and geometric computations, but the 3D display converts important operations from challenging geometric problem-solving and mental reconstruction to interactive visualizations of objects amongst terrain.

References

[1] http://www.anss.org/, "Advanced national seismic system," May 2005.

[2] P. L. Edsell, "Single-frequency fiber lasers," *Photonics Spectra*, pp. 52–54, Jan. 2005.

[3] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin, "Habitat monitoring with sensor networks," *Communications of the ACM*, vol. 47, no. 6, pp. 34–40, June 2004.

[4] P. Padhy, K. Martinez, A. Riddoch, H. L. R. Ong, and J. K. Hart, "Glacial environment monitoring using sensor networks," in *Proceedings of Real-World Wireless Sensor Networks (REALWSN'05)*, June 2005.

[5] C. D. Kidd, R. Orr, G. D. Abowd, C. G. Atkeson, I. A. Essa, B. MacIntyre, E. Mynatt, T. E. Starner, and W. Newstetter, "The aware home: A living laboratory for ubiquitous computing research," in *Cooperative Buildings*, 1999, pp. 191–198.

[6] D. L. Estrin et al., Ed., *Embedded, Everywhere*. National Academy Press, 2001.

[7] J. Gehrke and S. Madden, "Query processing in sensor networks," *IEEE Pervasive Computing*, vol. 3, no. 1, pp. 46–55, January–March 2004.

[8] A. Woo, S. Madden, and R. Govindan, "Networking support for query processing in sensor networks," *Communications of the ACM*, vol. 47, no. 6, pp. 47–52, June 2004.

[9] U.S. Marine Corps Combat Development Command, "Remote sensor operations," 1997.

[10] M. A. Livingston, A. Gregory, and W. B. Culbertson, "Camera control in three dimensions with a two-dimensional input device," *Journal of Graphics Tools*, vol. 5, no. 4, pp. 13–24, 2000.

[11] A. Appel, "Some techniques for shading machine rendering of solids," in *Proceedings of the AFIPS Spring Joint Computer Conference*, vol. 32, 1968, pp. 37–45.

[12] J. E. Warnock, "A hidden-surface algorithm for computer generated halftone pictures," Computer Science Department, University of Utah, Tech. Rep. TR 4–15, NTIS AD–753 671, June 1969.

[13] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice (2^{nd} ed.)*. Reading, MA: Addison-Wesley, 1990.

[14] G. S. Schmidt, S.-L. Chen, A. N. Bryden, M. A. Livingston, L. J. Rosenblum, and B. R. Osborn, "Multidimensional visual representations for underwater environmental uncertainty," *IEEE Computer Graphics and Applications*, vol. 24, no. 5, pp. 56–65, September/October 2004.